What is claimed is:

1. A method for updating a running process, comprising:

allocating in executable program code text first memory space operable to receive new program instructions comprising at least a first updated function;

allocating in executable program code text second memory space operable to receive address information related to said new program instructions;

running said executable program code;

stopping execution of said executable program code;

injecting a jump instruction and an address of said update table at a location in memory containing a first instruction of a first replaced function, wherein said address of said update table contains an address of a first instruction of said first updated function; and

resuming execution of said executable program code, wherein said first updated function is called in place of said first function, and wherein said executable code is updated in said memory.

2. The method of Claim 1, wherein said step of resuming execution of said executable program code comprises running an intermediate executable, wherein said intermediate executable comprises said updated copy of said executable program code in said memory.

3. The method of Claim 1, further comprising:

before said step of running said executable program code, copying said executable program code to said memory.

4. The method of Claim 1, further comprising:

injecting a jump instruction and an address of said update table at a location in a stored copy of said executable program code containing an address of said first function; and

populating said update table with an address of said first updated function, wherein a stored copy of said executable program code is updated.

5. The method of Claim 4, wherein said updated stored copy of said executable program code comprises final updated executable program code.

6. The method of Claim 1, further comprising:

determining a distance between a position within said code text at which execution of said executable program code is stopped and an address of a first function, wherein said first function is a function to be updated; and

in response to said first distance exceeding a predetermined amount, populating an update table stored in memory with an address of a first updated function.

7. The method of Claim 1, wherein said injected jump instruction replaces a first instruction of said first replaced function.

8. The method of Claim 1, wherein said predetermined amount is 8 bytes.

9. The method of Claim 1, wherein said second memory space is read only memory space.

10. A computational component for performing a method, the method comprising:

receiving information identifying:

a running executable program to be patched; and

a function to be replaced;

accessing a symbol table for said executable program to be patched;

obtaining from said symbol table an address of said function to be replaced;

stopping execution of said running executable program to be patched;

injecting in said running executable program to be patched a jump instruction and an address of a new function, wherein said new function is executed in place of said function to be replaced, and wherein a patched version of said executable program is created in memory; and

resuming execution of said executable program, wherein said patched version of said executable program is executed.

11. The method of Claim 10, further comprising providing a jump table in memory, wherein said injecting in said running executable program to be patched a jump instruction and an address of a new function comprises replacing a first instruction line associated with said function to be replaced with an instruction to jump to a first address contained within said jump table, and wherein said first address contained within said jump table comprises an address of a first instruction line of said new function.

12. The method of Claim 11, wherein said first instruction line associated with said function to be replaced comprises a first instruction of said function to be replaced, wherein said first instruction is not an instruction to jump to another address, and wherein said first instruction of said function to be replaced is replaced by said instruction to jump to a first address.

13. The method of Claim 11, wherein said first instruction line associated with said function to be replaced comprises an instruction to jump to a second address contained within said jump table, and wherein said instruction to jump to a second address is replaced by said instruction to jump to a first address.

14. The method of Claim 10, further comprising:

determining a number of instructions between a location within said executable program at which said executable program is stopped and an address of said function to be replaced.

15. The method of Claim 10, further comprising:

in response to said determined number of instructions being at least as great as a first selected number, injecting in a stored copy of said running executable program to be patched said jump instruction and said address of said new function in place of said address of said function to be replaced, wherein a patched version of said executable program is created.

16. The method of Claim 10, wherein said computational component comprises a computer readable storage medium containing instructions for performing the method.

17. The method of Claim 10, wherein said computational component comprises a logic circuit.

18. A system for updating executable program code, comprising:

means for receiving information identifying existing executable program code;

means for receiving information identifying a function to be replaced;

means for stopping execution of said existing executable program code;

means for inserting a jump code and an address associated with a replacement function in place of an address of said function to be replaced in said existing executable program code, wherein updated executable program code is created; and

means for resuming execution of said executable program code, wherein said updated executable program code is executed.

19. The system of Claim 18, further comprising jump table means, wherein said address associated with a replacement function comprises an address within said jump table containing an address of a first instruction of said replacement function.

20. A system for updating executing program code, comprising:

a create patch tool operable to receive information identifying an executable program to be updated and a function to be replaced;

a patch tool operable to query an operating system for a process identifier associated with said identified executable program;

a debugging utility operable to stop execution of said executable program to be updated and to determine a position of an instruction pointer associated with said executable program to be updated; and

a signal handler tool operable to replace in memory an address of said function to be replaced with an address of a replacement function in response to said position of said instruction pointer being at least a predetermined distance from said address of said replacement function, wherein said replacement function is executed instead of said function to be replaced upon resuming execution of said executable program, wherein said executable program is updated.

21. The system of Claim 20, wherein said signal handler is additionally operable to replace in a stored copy of said executable program an address of said function to be replaced with an address of a replacement function, wherein said stored copy of said executable program is updated.